INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2005; **00**:1–6 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

An assumed-gradient finite element method for the level set equation

Hashem M. Mourad[1], John Dolbow[1,*], Krishna Garikipati[2]

[1] *Department of Civil and Environmental Engineering, Duke University, Durham, North Carolina 27708, USA*
[2] *Department of Mechanical Engineering, University of Michigan, Ann Arbor, Michigan 48109, USA*

SUMMARY

The level set equation is a nonlinear advection equation, and standard finite-element and finite-difference strategies typically employ spatial stabilization techniques to suppress spurious oscillations in the numerical solution. We recast the level set equation in a simpler form by assuming that the level set function remains a signed distance to the front/interface being captured. As with the original level set equation, the use of an extensional velocity helps maintain this signed-distance function. For some interface-evolution problems, this approach reduces the original level set equation to an ordinary differential equation that is almost trivial to solve. Further, we find that sufficient accuracy is available through a standard Galerkin formulation without any stabilization or discontinuity-capturing terms. Several numerical experiments are conducted to assess the ability of the proposed assumed-gradient level set method to capture the correct solution, particularly in the presence of discontinuities in the extensional velocity or level-set gradient. We examine the convergence properties of the method and its performance in problems where the simplified level set equation takes the form of a Hamilton-Jacobi equation with convex/non-convex Hamiltonian. Importantly, discretizations based on structured and unstructured finite-element meshes of bilinear quadrilateral and linear triangular elements are shown to perform equally well. Copyright © 2005 John Wiley & Sons, Ltd.

KEY WORDS: Level set method; finite element; stabilization

1. INTRODUCTION

The standard level set equation (Osher and Sethian [12]) describes the evolution of a scalar function, ϕ, through

$$\dot{\phi} + v\|\boldsymbol{\nabla}\phi\| = 0, \tag{1}$$

where $v(\boldsymbol{x}, t)$ is the advection velocity. This partial differential equation is often used to describe the motion of an interface by associating its geometry with the zero-iso contour of ϕ. Typically,

*Correspondence to: John Dolbow, Department of Civil and Environmental Engineering, Duke University, Box 90287, Durham, NC 27708-0287, USA

Contract/grant sponsor: Sandia National Laboratories; contract/grant number: 184592

Received 16 March 2005
Revised 16 May 2005

ϕ is initialized as the signed distance to the interface, satisfying $\|\boldsymbol{\nabla}\phi\| = 1$. Signed-distance functions possess a number of desirable properties, and the advection velocity v is often replaced by a corresponding *extensional* velocity v_{e} such that $\|\boldsymbol{\nabla}\phi\| = 1$ is maintained. In this work, we examine the advantages of assuming, from the outset, that a signed-distance representation is maintained, allowing the above to be recast as

$$\dot{\phi} + v_{\mathrm{e}} = 0. \tag{2}$$

This is generally simpler than (1), and reduces to an ordinary differential equation when v_{e} is independent of $\boldsymbol{\nabla}\phi$. We explore the applicability of these ideas to cases where discontinuities in $\boldsymbol{\nabla}\phi$ and v_{e} are present, and examine discretizations of (2) based on structured and unstructured finite-element meshes.

The level set equation has been widely used in the computational science field as a mechanism for simplifying the representation of propagating interfaces (for example, see Osher and Fedkiw [13], Sethian [17]). Most discretizations of (1) have been based on finite-difference approximations over structured Cartesian meshes [12, 17], with implementations on triangulated domains first presented by Barth and Sethian [1]. The latter also considered stabilized Petrov-Galerkin finite-element approximations to (1) based on the work of Johnson [9]. Since these stabilized schemes did not control solution oscillations near discontinuities in $\boldsymbol{\nabla}\phi$, the use of the discontinuity-capturing operator advocated by Hansbo [7] was also studied in [1]. This nonlinear artificial-viscosity operator introduces a number of free parameters. Effective numerical solution schemes that do not require free parameters are obviously desirable.

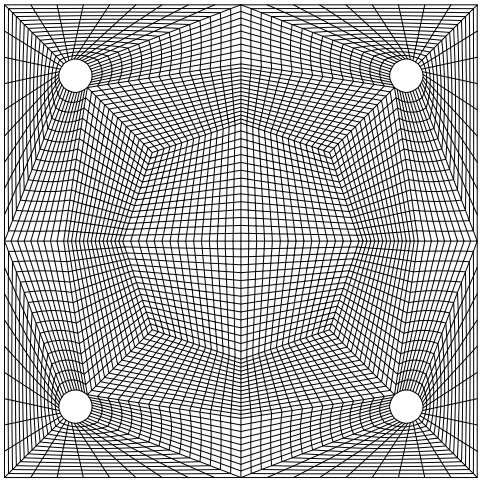


Figure 1. An example of a finite-element mesh based on arbitrary quadrilateral elements.

Further, for the finite-element analyst seeking to discretize (1) using a finite-element mesh based on arbitrary quadrilateral or brick elements (e.g. see Figure 1), relatively few options are available. Such meshes are popular for simulating structural mechanics problems for a number of reasons, most notably the increase in accuracy they provide over comparable

triangulated meshes. In principle, stabilized discretizations of (1) for arbitrary quadrilateral or brick elements could be developed using a Petrov-Galerkin formulation. To our knowledge, however, this has not been attempted.

In their work on grain-boundary migration, Mourad and Garikipati [11] recently introduced a simplified algorithm for the level set equation that did not employ stabilization. The algorithm can be derived from a straightforward Bubnov-Galerkin approximation to (2) with subsequent "mass lumping". Mourad and Garikipati [11] examined the method using structured Cartesian meshes of bilinear quadrilateral elements. This method was then employed by Ji *et al.* [8] for representing the evolution of phase boundaries over unstructured finite-element meshes. In the present paper, we examine the applicability of this method to a broader class of interface-evolution problems characterized by non-smooth solutions. We also examine the accuracy, convergence, and stability properties of the algorithm. Several numerical examples are presented to illustrate the robustness of the method, especially in cases where (2) takes the form of a Hamilton-Jacobi equation with convex or non-convex Hamiltonian.

This paper is organized as follows. In Section 2, we discuss the original level set equation and the justification for the modified form. We also provide an equivalent variational formulation. In Section 3 we provide details of the finite-element discretization and describe the numerical algorithms used in constructing the extensional velocity and in reinitializing the level set field. Numerical examples are then provided in Section 4. Finally, a summary and some concluding remarks are provided in Section 5.

2. PROBLEM FORMULATION

We consider a moving interface, $\mathscr{C}$, which divides the domain of interest, $\mathscr{B}$, into two disjoint open subsets, $\mathscr{B}^-(t)$ and $\mathscr{B}^+(t)$, for each t in some time-interval $\mathscr{T}$ of interest. This situation is depicted in Figure 2. An Eulerian formulation is sought for $\mathscr{C}(t)$ evolving with velocity v along its local normal, $\mathbf{n}_c$.

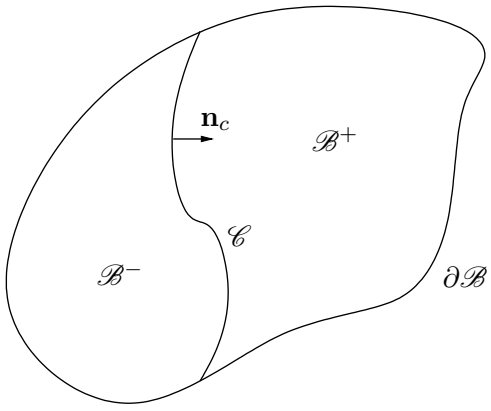


Figure 2. Schematic showing the domain of the problem, divided into regions $\mathscr{B}^+$ and $\mathscr{B}^-$ by the moving interface, $\mathscr{C}$. The normal to the interface, $\mathbf{n}_c$, is defined such that it always points into the $\mathscr{B}^+$ subdomain, as shown.

2.1. The level set equation

In the level set method, originally introduced by Osher and Sethian [12], $\mathscr{C}$ is parameterized as the zero level set,

$$\mathscr{C}(t) = \{\boldsymbol{x} : \ \phi(\boldsymbol{x}, t) = 0\}, \tag{3}$$

of the scalar function, ϕ, whose evolution is governed by

$$\dot{\phi} + v\|\boldsymbol{\nabla}\phi\| = 0, \qquad \forall\, (\boldsymbol{x}, t) \in \mathscr{B} \times \mathscr{T}, \tag{4}$$

where the normal velocity field $v = \mathrm{v}$ on $\mathscr{C}$, and $\|\cdot\|$ denotes the magnitude of the vector. The signed-distance function to the interface is often used as the initial condition; i.e.

$$\phi(\boldsymbol{x}, 0) = \begin{cases} \min\limits_{\mathbf{p}\in\mathscr{C}} \|\boldsymbol{x} - \mathbf{p}\|, & \forall\, \boldsymbol{x} \in \mathscr{B}^+, \\ -\min\limits_{\mathbf{p}\in\mathscr{C}} \|\boldsymbol{x} - \mathbf{p}\|, & \forall\, \boldsymbol{x} \in \mathscr{B}^-, \end{cases} \tag{5}$$

and the velocity, v, is replaced by a corresponding extensional velocity, v_e, which satisfies $v_\mathrm{e} = v = \mathrm{v}$ on $\mathscr{C}$, as well as

$$\boldsymbol{\nabla}\phi \cdot \boldsymbol{\nabla} v_\mathrm{e} = 0, \qquad \forall\, \boldsymbol{x} \in \mathscr{B}. \tag{6}$$

For sufficiently smooth ϕ and v_e, this preserves the initial signed-distance function, as shown by Zhao *et al.* [21]:

$$\begin{aligned} \frac{\partial}{\partial t}\|\boldsymbol{\nabla}\phi\|^2 &= 2\boldsymbol{\nabla}\phi \cdot \frac{\partial}{\partial t}\boldsymbol{\nabla}\phi \\ &= 2\boldsymbol{\nabla}\phi \cdot \boldsymbol{\nabla}\dot{\phi} \\ &= -2\boldsymbol{\nabla}\phi \cdot \boldsymbol{\nabla}(v_\mathrm{e}\|\boldsymbol{\nabla}\phi\|) \\ &= -2(\boldsymbol{\nabla}\phi \cdot \boldsymbol{\nabla} v_\mathrm{e})\|\boldsymbol{\nabla}\phi\| - 2\boldsymbol{\nabla}\phi \cdot v_\mathrm{e}(\boldsymbol{\nabla}\|\boldsymbol{\nabla}\phi\|), \end{aligned} \tag{7}$$

which vanishes by virtue of (6) and since a distance function satisfies $\|\boldsymbol{\nabla}\phi\| = 1$ and $\boldsymbol{\nabla}\|\boldsymbol{\nabla}\phi\| = \mathbf{0}$, by definition. Signed distances to the interface are preferred because a uniform separation of the level sets leads to accurate approximations of the local unit normal, $\boldsymbol{n}$, and curvature, κ, via

$$\boldsymbol{n} = \frac{\boldsymbol{\nabla}\phi}{\|\boldsymbol{\nabla}\phi\|}, \tag{8}$$

$$\kappa = \boldsymbol{\nabla} \cdot \boldsymbol{n} = \boldsymbol{\nabla} \cdot \frac{\boldsymbol{\nabla}\phi}{\|\boldsymbol{\nabla}\phi\|}. \tag{9}$$

2.2. Modified level set equation and the effect of discontinuities

Based on the arguments of Section 2.1, Mourad and Garikipati [11] assume that $\|\boldsymbol{\nabla}\phi\| = 1$ is maintained everywhere in $\mathscr{B}$, for all $t \geq 0$, and hence replace (4) by

$$\dot{\phi} + v_\mathrm{e} = 0, \qquad \forall\, (\boldsymbol{x}, t) \in \mathscr{B} \times \mathscr{T}. \tag{10}$$

We note that in those cases where v_e is independent of all derivatives of ϕ with respect to $\boldsymbol{x}$, the above is an ordinary differential equation.

To gain some insight into this approach, consider the following one-dimensional example. We consider the domain $\mathscr{B} = \{x : x \in (0, 1)\}$ and write x_s for the position of the interface. Assume

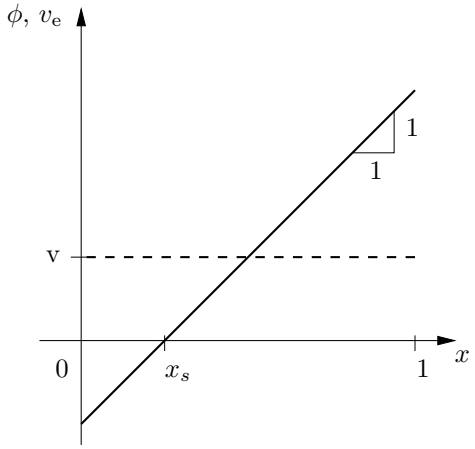


Figure 3. One-dimensional example of the level set function ϕ (solid line) as a signed distance to the interface, and an extensional velocity (dashed line).

the level set function is initially a signed distance to the interface, i.e. $\phi(x,0) = x - x_s|_{t=0}$. For concreteness, such a function is shown in Figure 3. The only function satisfying (6) in this simple example is $v_e = \mathrm{v}$, a constant field.

Under these conditions, the modified level set equation (10) simply implies that ϕ decreases monotonically with time (assuming v is positive), and further that the interface evolves with the correct speed v. One way to view the original level set equation (4), is to recognize the term $\|\boldsymbol{\nabla}\phi\|$ as a "correction" to the normal velocity field, such that the zero level contour of ϕ evolves at the correct speed even when ϕ is *not* a signed distance to the interface.

While relatively trivial, the above analogy does hold for multiple dimensions. Following a procedure analogous to $(7)_{1-4}$, except using the modified equation (10), we obtain

$$\frac{\partial}{\partial t}\|\boldsymbol{\nabla}\phi\|^2 = -2\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}v_e. \tag{11}$$

This is a simpler result than $(7)_4$, but the implication is the same: the solution to (10) remains a signed-distance function. Further, we note that in contrast to $(7)_4$, a term proportional to $\boldsymbol{\nabla}\|\boldsymbol{\nabla}\phi\|$ does not appear in (11). This suggests that the assumed-gradient algorithm may be less sensitive to perturbations in the initial signed-distance function.

Both $(7)_4$ and (11) were obtained by assuming that the level set function and extensional velocity remain sufficiently smooth. However, in general, $\boldsymbol{\nabla}\phi$ and v_e may not be continuous on $\mathscr{B}$. For instance, consider the case in which the extensional velocity takes the form

$$v_e = v_1 H_z + v_2(1 - H_z), \tag{12}$$

with v_1 and v_2 sufficiently smooth and H_z the Heaviside unit-step function associated with some surface $\mathscr{Z}$ (independent of $\mathscr{C}$) in the domain. Further, assume that ϕ is smooth on $\mathscr{B}$ and

$$\boldsymbol{\nabla}v_1\cdot\boldsymbol{\nabla}\phi = \boldsymbol{\nabla}v_2\cdot\boldsymbol{\nabla}\phi = 0, \qquad \forall\ \boldsymbol{x}\in\mathscr{B}. \tag{13}$$

Thus, v_e satisfies (6) everywhere except on $\mathscr{Z}$ where $\boldsymbol{\nabla}v_e$ is singular. Again following a

procedure similar to $(7)_{1-4}$, and using (10), we obtain

$$\begin{aligned}\frac{\partial}{\partial t}\|\boldsymbol{\nabla}\phi\|^2 &= 2\boldsymbol{\nabla}\phi\cdot\frac{\partial}{\partial t}\boldsymbol{\nabla}\phi\\ &= 2\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}\dot{\phi}\\ &= -2\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}v_{\mathrm{e}}\\ &= -2(\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}v_1)H_{\mathrm{z}} - 2(\boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}v_2)(1-H_{\mathrm{z}}) - 2(\boldsymbol{\nabla}\phi\cdot\mathbf{n}_{\mathrm{z}})(v_1-v_2)\delta_z,\end{aligned} \tag{14}$$

where $\mathbf{n}_{\mathrm{z}}$ is the unit normal to the surface $\mathscr{Z}$ and δ_z is the Dirac delta function associated with that surface, i.e. $\delta_z\mathbf{n}_{\mathrm{z}} = \boldsymbol{\nabla}H_{\mathrm{z}}$, in the sense of distributions. The first two terms on the right of $(14)_4$ vanish in light of (13), but the last term will not vanish in general. We therefore expect the level set field to not remain a signed distance to the interface. We will examine the consequences of this result in Section 4, with the aid of several numerical examples.

2.3. Variational formulation

Writing $\mathscr{U}$ for the space of admissible level set fields, the variational equivalent to (10) is given by: Find $\phi \in \mathscr{U}$ such that

$$\int_{\mathscr{B}} \dot{\phi} w \,\mathrm{dV} = -\int_{\mathscr{B}} v_{\mathrm{e}} w \,\mathrm{dV} \tag{15}$$

for all $w \in \mathscr{U}$.

3. DISCRETIZATION AND ALGORITHMS

Finite-element computations entail the projection of the solution space, $\mathscr{U}$, and the associated space of variations onto the finite-dimensional subspace $\mathscr{U}^h$. We use M to denote the total number of elements in the mesh and consider a regular finite-element partition $\mathscr{B}^h = \bigcup_{e=1}^{M} \mathscr{B}^e$, with $\mathscr{B}^h = \mathscr{B}$ but with element edges chosen, generally, to be independent of the interface geometry. Letting $P^j(\mathscr{B}^e)$ denote the space of complete polynomials of order less than or equal to j over element $\mathscr{B}^e$, we introduce

$$\{N_i \in C^0(\mathscr{B}^h) : \ N_i|_{\mathscr{B}^e} \in P^j(\mathscr{B}^e)\}, \tag{16}$$

where N_i, with $i = 1, 2, \ldots$, are the nodal shape functions. Thereupon, we approximate ϕ using

$$\phi^h(\boldsymbol{x}, t) = \sum_{i\in I} N_i(\boldsymbol{x}) d_i(t), \tag{17}$$

where I denotes the set of all nodes in the mesh.

We consider the solution on the time interval $\mathscr{T} = [0, t_f]$, partitioned into time steps as $[t_n, t_{n+1}]$. We approximate $\dot{\phi}$ using a finite-difference stencil in time. For example, when a forward-Euler scheme is used to approximate $\dot{\phi}$, the weak form of the problem is stated as: Find $\phi^h_{n+1} \in \mathscr{U}^h$ such that

$$\int_{\mathscr{B}} \phi^h_{n+1} w^h \,\mathrm{dV} = \int_{\mathscr{B}} \phi^h_n w^h \,\mathrm{dV} - \Delta t \int_{\mathscr{B}} (v^h_{\mathrm{e}})_n w^h \,\mathrm{dV}, \tag{18}$$

for all $w^h \in \mathscr{U}^h$, where $(\cdot)_n$ denotes a time-discrete quantity evaluated at $t = t_n$.

A second-order Runge-Kutta scheme yields a first stage given by: Find $\phi^h_{n+\frac{1}{2}} \in \mathscr{U}^h$ such that

$$\int_{\mathscr{B}} \phi^h_{n+\frac{1}{2}} w^h \,\mathrm{dV} = \int_{\mathscr{B}} \phi^h_n w^h \,\mathrm{dV} - \Delta t \int_{\mathscr{B}} (v^h_{\mathrm{e}})_n w^h \,\mathrm{dV}, \tag{19}$$

for all $w^h \in \mathscr{U}^h$. This is followed by a second stage: Find $\phi^h_{n+1} \in \mathscr{U}^h$ such that

$$\int_{\mathscr{B}} \phi^h_{n+1} w^h \,\mathrm{dV} = \frac{1}{2} \int_{\mathscr{B}} \left(\phi^h_n + \phi^h_{n+\frac{1}{2}} \right) w^h \,\mathrm{dV} - \frac{\Delta t}{2} \int_{\mathscr{B}} (v^h_{\mathrm{e}})_{n+\frac{1}{2}} w^h \,\mathrm{dV}. \tag{20}$$

Both first-order and second-order schemes will be examined in Section 4.

3.1. Mass lumping

Substituting (17) (and an analogous expansion for w^h) into (18) and lumping quantities at the nodes yields the simple update formula

$$(d_i)_{n+1} = (d_i)_n - \Delta t \; (v^h_{\mathrm{e}})_n(\boldsymbol{x}_i). \tag{21}$$

This first-order scheme was used successfully by Mourad and Garikipati [11]. The following second-order accurate variation is obtained by substituting (17) into (19) and (20), and lumping quantities at the nodes:

$$(d_i)_{n+\frac{1}{2}} = (d_i)_n - \Delta t \; (v^h_{\mathrm{e}})_n(\boldsymbol{x}_i), \tag{22a}$$

$$(d_i)_{n+1} = \frac{1}{2} \left((d_i)_n + (d_i)_{n+\frac{1}{2}} \right) - \frac{\Delta t}{2} \; (v^h_{\mathrm{e}})_{n+\frac{1}{2}}(\boldsymbol{x}_i). \tag{22b}$$

3.2. Construction of approximate extensional velocity field

The goal here is to construct the approximation v^h_{e} to the extensional velocity. A standard approach [17] is to construct v^h_{e} such that

$$v^h_{\mathrm{e}}|_{\phi=0} = \mathrm{v}^h \qquad \text{and} \qquad \boldsymbol{\nabla} v^h_{\mathrm{e}} \cdot \boldsymbol{\nabla} \phi^h = 0, \tag{23}$$

taking advantage of fast marching methods in the latter. A simpler, albeit less efficient approach originally suggested by Malladi *et al.* [10] and previously used in [6, 11] is followed here. First, we determine the closest-point projection, $\mathbf{p}$, of each node $\boldsymbol{x}_i$ onto the interface as represented by the set of subsurfaces $\{\mathscr{C}^e\}$, where

$$\mathscr{C}^e = \left\{ \boldsymbol{x} \in \mathscr{B}^e : \sum_{i \in I} N_i(\boldsymbol{x}) d_i = 0 \right\}. \tag{24}$$

The algorithm used for this purpose is described in Section 3.3 below. The extensional velocity at each node is then assigned via

$$v^h_{\mathrm{e}}(\boldsymbol{x}_i) = \mathrm{v}^h(\mathbf{p}(\boldsymbol{x}_i)). \tag{25}$$

We assume the normal velocity v to be a given function of position or geometry of the interface. The approximation v^h is therefore obtained from $\mathrm{v}(\cdot)$ by replacing the arguments by their discrete counterparts. When the normal velocity of the interface is a function of the interfacial normal, we have found it advantageous to consistently use the assumption $\|\boldsymbol{\nabla}\phi\| = 1$ when approximating the normal through (8). The importance of consistently using this assumption will be examined in Section 4.

3.3. Reinitialization of the level set field

The numerical strategy given by (18) or (19)–(20) does not insure that $\|\boldsymbol{\nabla}\phi^h\| = 1$ is maintained. Due to both temporal and spatial errors in the discretization, we anticipate that $\|\boldsymbol{\nabla}\phi^h\|$ may deviate sufficiently far from unity such that (18) or (19)–(20) cease to be sufficiently accurate in describing the interface evolution. It is therefore useful to consider reinitialization algorithms that effectively reset the level set field to an approximate signed-distance function.

Sethian [17] uses the fast marching method to reinitialize ϕ^h as a signed-distance function to the interface while simultaneously constructing an extensional velocity. Other re-distancing algorithms include the iterative strategy proposed by Sussman and Fatemi [19]. Here, we follow the strategy described by Mourad and Garikipati [11] for reinitialization. The algorithm consists of a loop over the set of all nodes, $\{\boldsymbol{x}_i\}$, in which the nodal values of the level set variable are reinitialized as

$$d_i = \|\boldsymbol{x}_i - \mathbf{p}(\boldsymbol{x}_i)\| \ \text{sign}\Big(\big(\boldsymbol{x}_i - \mathbf{p}(\boldsymbol{x}_i)\big) \cdot \mathbf{n}_c\big(\mathbf{p}(\boldsymbol{x}_i)\big)\Big). \tag{26}$$

The algorithm in Box 1 is used to locate the closest-point projection, $\mathbf{p}$, of each node $\boldsymbol{x}_i$ onto the interface as represented by the set of subsurfaces $\{\mathscr{C}^e\}$. To simplify the implementation of this algorithm, these subsurfaces are assumed to be planar (rectilinear in the two-dimensional case). In other words, the algorithm searches for the closest-point projection onto $\widetilde{\mathscr{C}^e}$ which is a simplified representation of $\mathscr{C}^e$; in a two-dimensional framework, it is the rectilinear segment passing through the endpoints of $\mathscr{C}^e$ [see Figure 4(a)].

```
FOR each node, x_i, DO
  ASSIGN p(x_i) ← ⟨∞, ∞⟩^T
ENDDO
FOR each subsurface, C^e, DO
  FOR each node, x_i, DO
    FIND point p^e(x_i) such that ||x_i − p^e(x_i)|| = min_{z∈C̃^e} ||x_i − z||
    IF ||x_i − p^e(x_i)|| < ||x_i − p(x_i)|| THEN
      ASSIGN p(x_i) ← p^e(x_i)
    ENDIF
  ENDDO
ENDDO
```

Box 1. The algorithm used to locate the closest-point projection of each node on the interface.

Naturally, the error introduced by this simplifying assumption is less significant when the interface is relatively smooth and the mesh is sufficiently refined. Further, this assumption is exact in linear triangular elements but introduces some error in bilinear quadrilateral elements. To illustrate this notion, the representation of a curved interface passing through the square domain, $\mathscr{B} = \{\boldsymbol{x} : \boldsymbol{x} \in (-1,1) \times (-1,1)\}$, is shown in Figure 4. When $\mathscr{B}$ is discretized using one bilinear quadrilateral element, the interface is represented by a single *curved* subsurface, $\mathscr{C}^e$, as shown in Figure 4(a). Error is therefore introduced when the planar (rectilinear) $\widetilde{\mathscr{C}^e}$ is used in lieu of $\mathscr{C}^e$ for the purpose of locating closest-point projections. By contrast, it can

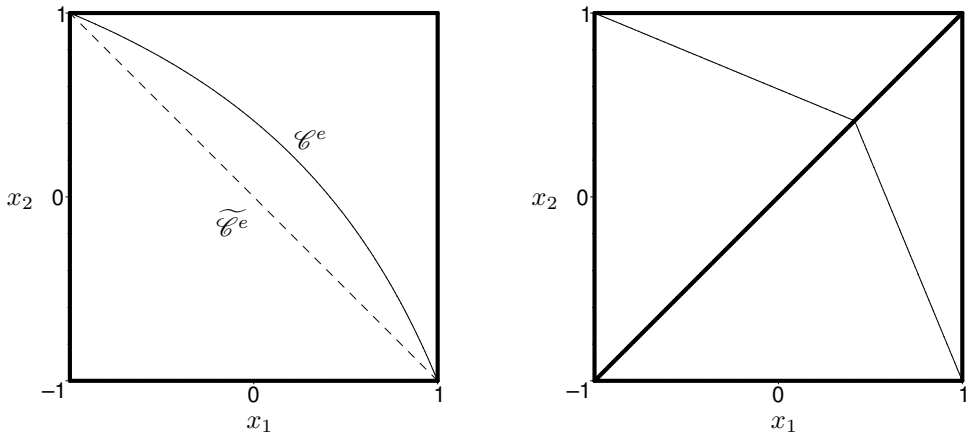


Figure 4. A curved interface is represented (a) by one curved subsurface when the mesh consists of a single bilinear quadrilateral element, or (b) by two planar (rectilinear) subsurfaces when the domain is discretized using two linear triangular elements.

be seen from Figure 4(b) that when $\mathscr{B}$ is discretized using two linear triangular elements, the same interface—i.e. the interface implied by the same nodal values of ϕ^h—is represented by two planar (rectilinear) subsurfaces. In this case, $\widetilde{\mathscr{C}}^e \equiv \mathscr{C}^e$.

It is noted that the method of finding closest-point projections outlined in Box 1 is relatively expensive. For L subsurfaces and N nodes in the mesh, the complexity of the algorithm is at best $\mathcal{O}(L \times N)$.[†] By comparison, fast-marching methods for reinitialization have $\mathcal{O}(N \log N)$ complexity and are much more efficient [17]. Regardless, it is important to resort to reinitialization only when it is needed. It must also be noted however that in problems where the closest-point projections must be determined for the purpose of constructing the extensional velocity field (see Section 3.2 above), reinitialization consists only of performing the assignment operation (26) once for each node, and hence does not incur a significant additional computational cost.

3.4. Computing the local curvature of the interface

Using the assumption $\|\boldsymbol{\nabla}\phi\| = 1$, Equations (8)–(9) can be reduced to

$$\boldsymbol{n} = \boldsymbol{\nabla}\phi, \tag{27}$$

$$\kappa = \boldsymbol{\nabla} \cdot \boldsymbol{n} = \nabla^2\phi. \tag{28}$$

[†]This simple estimate neglects the cost of constructing the subsurfaces in the first place, and determining the closest point projection on each, both of which obviously incur additional expense.

However, the curvature cannot be approximated as $\kappa^h = \nabla^2 \phi^h$ with linear or bilinear elements, where $\nabla^2 N_i = 0$, and consequently also $\nabla^2 \phi^h = \sum_{i \in I} (\nabla^2 N_i) d_i = 0$, in element interiors.

To overcome this difficulty, we introduce the vector field $\tilde{\boldsymbol{n}}^h = \sum_{i \in I} N_i \boldsymbol{g}_i$ which minimizes

$$\int_{\mathscr{B}} \| \tilde{\boldsymbol{n}}^h - \boldsymbol{\nabla} \phi^h \|^2 \, \mathrm{dV}.$$

This least squares problem can be expressed as: Find $\boldsymbol{g}_i$ such that

$$\sum_{i \in I} \int_{\mathscr{B}} N_j N_i \, \mathrm{dV} \, \boldsymbol{g}_i = \int_{\mathscr{B}} N_j \boldsymbol{\nabla} \phi^h \, \mathrm{dV}, \tag{29}$$

Lumping quantities at the nodes yields the simple expression

$$\boldsymbol{g}_i = \frac{\displaystyle\int_{\mathscr{B}} N_i \boldsymbol{\nabla} \phi^h \, \mathrm{dV}}{\displaystyle\int_{\mathscr{B}} N_i \, \mathrm{dV}}. \tag{30}$$

Finally, the curvature is approximated as

$$\kappa^h = \boldsymbol{\nabla} \cdot \tilde{\boldsymbol{n}}^h. \tag{31}$$

This least-squares projection procedure was used successfully with bilinear quadrilateral elements by Dolbow *et al.* [5], Ji *et al.* [8], and Mourad and Garikipati [11]. A similar procedure for unstructured triangulated meshes was proposed by Barth and Sethian [1]. A procedure incorporating two least-squares projection operations was proposed for linear triangular elements by Chessa and Belytschko [3].

3.5. Stability

The critical time step Δt for the original level set equation is given by (Sethian [17])

$$\max_{\boldsymbol{x} \in \mathscr{B}^h} v_{\mathrm{e}}^h \Delta t \leq h, \tag{32}$$

where h is the characteristic mesh size. We have not performed a stability analysis of the modified level set equation, but it is clear that the stability of the scheme depends on the particular form of the extensional velocity. For example, for some extensional velocities the final governing equation is identical to that obtained via (4) and stability is governed by (32). By contrast, the scheme is unconditionally stable for constant extensional velocities. We will examine examples of both cases in the next section. More generally, we have found time step sizes of

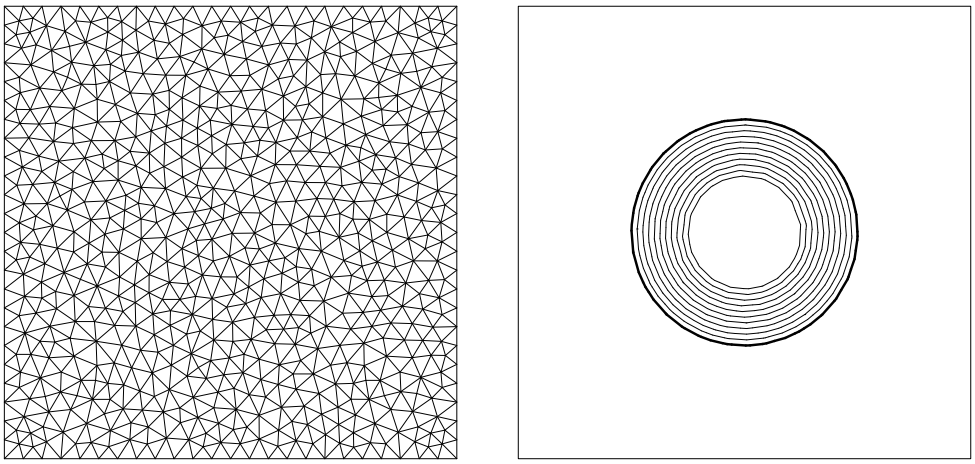
$$\Delta t = \frac{1}{2} \frac{h}{\displaystyle\max_{\boldsymbol{x} \in \mathscr{B}^h} v_{\mathrm{e}}^h} \tag{33}$$

to yield stable results that are also sufficiently accurate temporally. This choice is used in all of the examples in Section 4, unless otherwise noted.

4. NUMERICAL EXAMPLES

In this section, we examine the performance of the assumed-gradient level set algorithm for a series of problems, using finite-element discretizations based on both bilinear quadrilateral and linear triangular elements. Unless otherwise noted, results are reported using the second-order Bubnov-Galerkin (19)-(20) method with consistent mass matrix, without reinitialization, and with time-step size given by (33).

4.1. Circle collapsing at unit speed



(a) (b)

Figure 5. First benchmark problem: (a) Coarsest triangular mesh used ($h = 0.16$). (b) Location of the zero level set, computed at equal intervals within $t \in [0, 0.5]$.

As a first benchmark problem, we consider a circular interface shrinking at unit speed; i.e. v = 1 everywhere on $\mathscr{C}$ with $\mathbf{n}_c$ pointing toward the center of the circle. Accordingly, the extensional velocity $v_e = 1$ is used, and the assumed-gradient equation (10) reduces to $\dot{\phi} + 1 = 0$. The calculations are carried out on a sequence of four Cartesian meshes with characteristic element size, h, successively reduced by a factor of two. This process is repeated with unstructured triangulated meshes. The computational domain, a 4×4 square, and the coarsest triangular mesh used ($h = h_{\max} = 0.16$) are shown in Figure 5(a). The level set field is initialized as a signed-distance function to a circular interface with radius $r_0 = 1$.

Since v_e is constant in this example, we note that the first-order (18) and second-order (19)-(20) algorithms are unconditionally stable and yield identical results. The time-step size $\Delta t = 10^{-3}$ is chosen arbitrarily and is used with all meshes. Figure 5(b) shows the numerical results obtained using the coarsest triangular mesh. The figure shows the location of the interface at equal intervals within $t \in [0, 0.5]$. For clarity, the initial zero level set is shown as a thicker line.

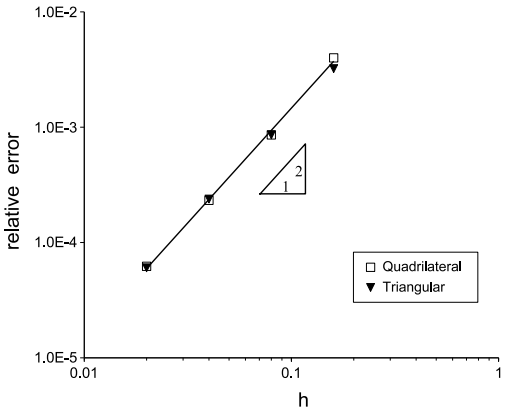


Figure 6. The error in the L^2-norm plotted against the characteristic element size for Cartesian and triangulated meshes.

To assess the accuracy of the numerical results obtained, the relative L^2-error in ϕ^h is computed for each value of h. The exact solution is a signed-distance function with a zero level set that remains circular with radius $r(t) = r_0 - t$, for $0 \leq t \leq r_0$. The plots of the relative error against h, shown in Figure 6, confirm that optimal second-order convergence is achieved on Cartesian as well as unstructured meshes. This result is not particularly surprising since the solution at $t = t_n = n\Delta t$ is given trivially by the initial data as $\phi^h(\boldsymbol{x}, t_n) = \phi^h(\boldsymbol{x}, 0) - n\Delta t$. The accuracy is thus entirely governed by the interpolation error in the initial signed distance function.

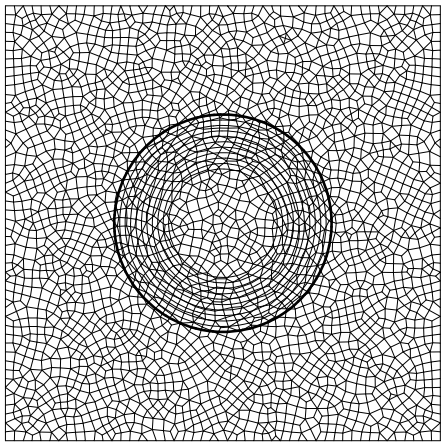
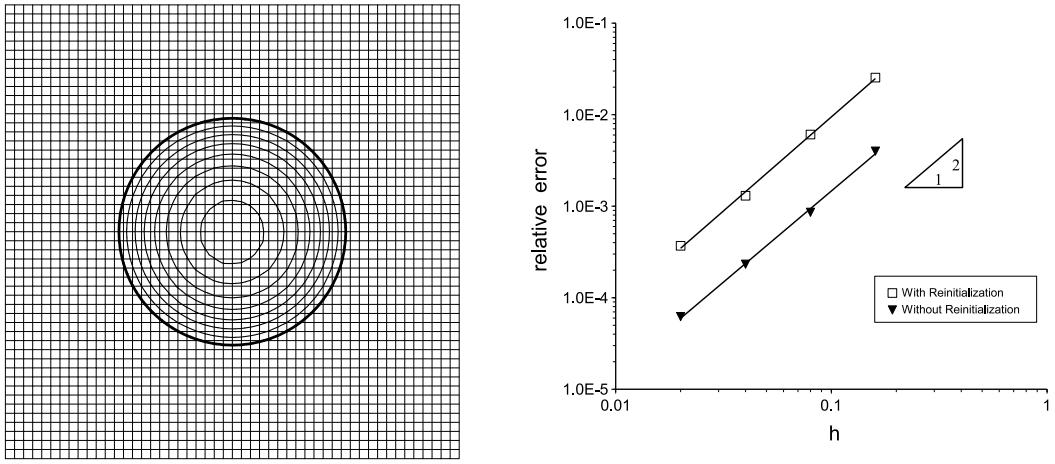


Figure 7. Computations carried out on a mixed unstructured mesh ($h = 0.08$). The location of the zero level set is shown at equal intervals within $t \in [0, 0.5]$.

It is noted that the choice of spatial discretization is not restricted to Cartesian grids and unstructured triangulations. For example, a structured or unstructured mesh comprising both

quadrilateral and triangular elements can be used. To illustrate this idea, the same problem is repeated on such a mixed unstructured mesh, shown in Figure 7. The numerical results obtained are also shown in the figure and do not exhibit any noticeable defects.

4.2. Circle collapsing at curvature-dependent speed



(a) (b)

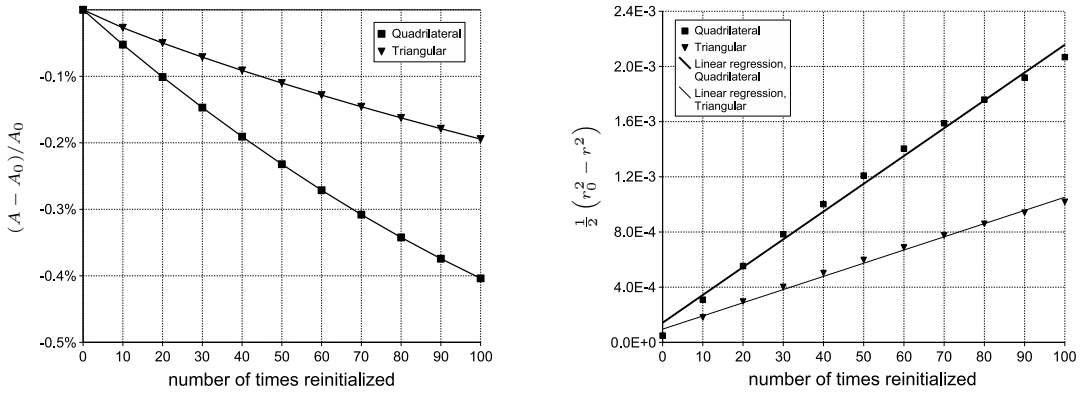
Figure 8. Second benchmark problem: (a) Numerical results obtained on the mesh shown ($h = 0.08$). The location of the zero level set is shown at equal intervals within $t \in [0, 0.455]$. (b) The error in the L^2-norm plotted against the characteristic element size. The error introduced by the reinitialization scheme can be seen clearly.

As a second benchmark problem, we consider the curvature-driven collapse of a circular interface. In this problem we have $\mathrm{v} = -\kappa$, where κ is the local curvature of the interface. It is noted that $\kappa < 0$ with $\mathbf{n}_c$ pointing toward the center of the circle, and vice versa. The curvature is evaluated as described in Section 3.4. The extensional velocity field, v_e, is constructed by closest-point projection as described in Section 3.2. A 4×4 square computational domain is used. The calculations are repeated on a sequence of four Cartesian meshes with characteristic element size, h, successively reduced by a factor of two. The computational results obtained with $h = 0.08$ are shown in Figure 8(a).

The exact solution is a signed-distance function with a circular zero level set whose radius is given by $r(t) = \sqrt{r_0^2 - 2t}$. The initial radius is $r_0 = 1$. The forward-Euler scheme (18) is used to advance the solution in time over the interval $t \in [0, 0.455]$. Based on the highest speed reached within this time interval, $\mathrm{v}_{\max} = 1/r|_{t=0.455} = 10/3$, and the smallest mesh size used, $h_{\min} = 0.02$, the time step size $\Delta t = 10^{-3}$ is chosen and is used with all meshes.

To quantify the error introduced by the reinitialization scheme of Section 3.3, the computations are repeated with reinitialization invoked at the end of every time step. The relative L^2-error in ϕ^h, with and without reinitialization, is plotted against h in Figure 8(b).

It is clear from this figure that optimal second-order convergence is achieved, regardless of whether reinitialization is used. The fact that it preserves the optimal convergence rate is a known advantage of the reinitialization scheme used (see Peng *et al.* [14]). However, it is also clear from Figure 8(b) that the error introduced by the reinitialization algorithm is not negligible; it is therefore important to avoid invoking it unnecessarily.



(a) (b)

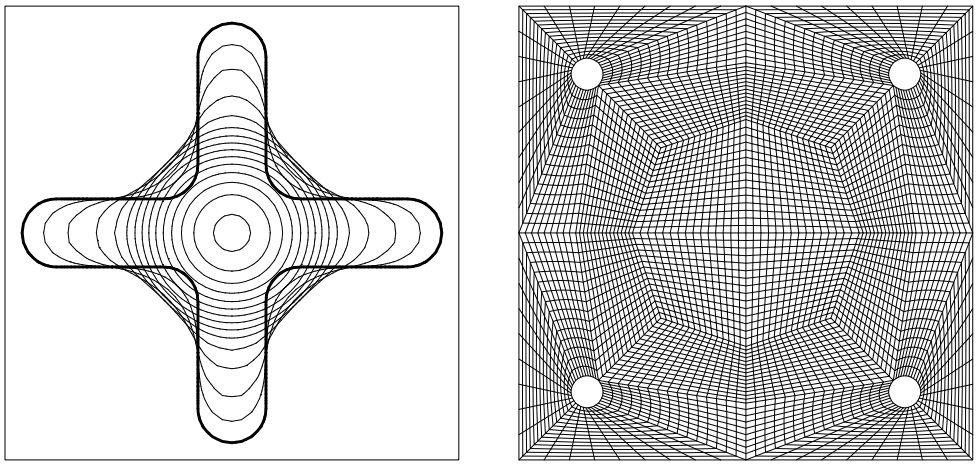
Figure 9. The plots shown demonstrate that (a) the area enclosed by the circular front is lost when reinitialization is performed using the scheme of Section 3.3, and that (b) every time the reinitialization scheme is executed, the interface moves a distance proportional to its curvature.

We also examine the capacity of the reinitialization scheme to conserve the area enclosed by the interface. For this purpose, we begin with a circular interface with $r_0 = 1$ and we repeatedly invoke the reinitialization algorithm at $t = 0$, i.e. without advancing the solution in time. This numerical experiment is conducted on both Cartesian and triangulated grids with $h = 0.02$. In Figure 9(a), the change in the area enclosed by the interface is plotted, as a percentage of the initial enclosed area, against the number of times the reinitialization algorithm is executed. Clearly, reinitialization leads to area loss with both types of grids. However, this area loss is less severe on triangulated grids, as expected (see Section 3.3). In addition, the plot in Figure 9(b), suggests that we can write $\frac{1}{2}\left(r_0^2 - r^2\right) = \epsilon N_r$, where r is the radius of the circular interface after reinitializing N_r times. This implies that

$$\frac{\mathrm{d}\, r}{\mathrm{d}N_r} = -\frac{\epsilon}{r} = -\epsilon\kappa, \tag{34}$$

i.e. every time it is executed, the reinitialization scheme moves the interface a distance proportional to its curvature. The value of the proportionality constant, ϵ, is determined via linear regression from the data in Figure 9(b). As expected, it is found to be larger with the Cartesian mesh ($\sim 2.02 \times 10^{-5}$) than with the triangulated grid ($\sim 9.56 \times 10^{-6}$). Periodic reinitialization can thus be expected to produce an effect similar to the addition of an artificial curvature-dependent term.

4.3. Non-convex curve collapsing at curvature-dependent speed



(a) (b)

Figure 10. Curvature-driven collapse of a non-convex curve: (a) Location of the zero level set, computed at equal intervals within $t \in [0, 0.56]$. (b) Non-Cartesian finite-element mesh used.

As another example of curvature flow, we examine the collapse of the star-shaped curve, shown in Figure 10(a). The original profile, shown as a thicker line, is clearly non-convex. Consequently, some parts of the curve propagate outwards initially, until the curve loses its non-convexity. Beyond this point, all parts of the curve propagate inwards, forming a circle which collapses as in the previous example. The mesh shown in Figure 10(b) is used in this problem to illustrate the possibility of carrying out the computations on arbitrary (non-Cartesian) meshes based on quadrilateral elements. The forward-Euler scheme (18) is used with $\Delta t = 10^{-3}$ to advance the solution in time over the interval $t \in [0, 0.56]$. Reinitialization is performed once every 10 time steps. Importantly, the results shown in Figure 10(a) do not appear to exhibit a sensitivity to the structure of the mesh.

This example demonstrates the applicability of the present algorithm to problems in which the velocity of the interface changes sign. It is noted that less expensive algorithms which rely on solving the Eikonal equation, e.g. the fast marching method [17] and ordered upwind methods [18], cannot be used with such problems.

4.4. Merging circles

Next, we provide an example problem involving changes in topology and discontinuities in v_e and $\boldsymbol{\nabla}\phi$. We consider a problem in which the initial interface consists of two disjoint circular fronts—the two innermost circles in Figure 11. The right front has an initial radius $r_0 = 0.5$ and expands with normal velocity $\mathrm{v} = 0.2$, while the left one is initially twice as large but expands twice as slowly. The boundary of the 8×4 rectangular computational domain is also

shown in Figure 11. We report results obtained on a uniform mesh of quadrilateral elements with characteristic size $h = 0.04$, using the first-order update formula (21) and time-step size given by (33). The lumped-mass approach is chosen in this case to illustrate the efficacy of this simple algorithm; qualitatively similar results are given by the first or second-order consistent mass algorithms.

```
IF x ≤ x_ℓ OR ( x_ℓ < x < x_r AND n_x > 0 ) THEN
  v(x) = 0.1
ELSE
  v(x) = 0.2
ENDIF
```

Box 2. The algorithm used to determine which front a point belongs to, and thus to select the correct local velocity.

During the simulation, the normal velocity at a point $\boldsymbol{x} \equiv \langle x, y \rangle^T$ on the zero level set is determined depending on its coordinates and on the direction of the local normal to the interface, $\boldsymbol{n} \equiv \langle n_x, n_y \rangle^T$. This is achieved with the aid of the algorithm in Box 2, where $\langle x_\ell, y_c \rangle^T$ and $\langle x_r, y_c \rangle^T$ are the center points of the left and right fronts, respectively. In the current numerical example, these parameters take the values $x_\ell = 2.5$, $x_r = 5.5$ and $y_c = 2.0$.

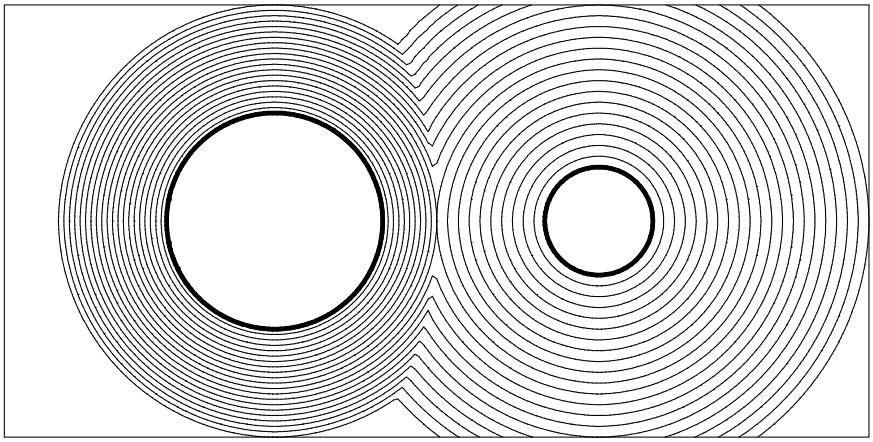


Figure 11. Two circular fronts growing at different speeds and merging. Location of the fronts is shown at equal intervals within $t \in [0, 10]$.

The level set field is initialized as the signed distance to the closest front, giving rise to a discontinuity in $\boldsymbol{\nabla}\phi^h$ at points which are equidistant from both fronts. After the two fronts merge, these discontinuities manifest themselves as sharp corners in the zero level set. In addition, the difference in propagation speeds leads to a discontinuity in $\boldsymbol{\nabla} v_{\mathrm{e}}^h$ when the extensional velocity field is constructed by closest-point projection (see Section 3.2) from the fronts. Nevertheless, it is clear from Figure 11 that the numerical solution is free from spurious effects, even in the neighborhood of these discontinuities.

4.5. Simulation of etching processes

Finally, we present numerical results from simulations of surface etching processes used in the manufacturing of semiconductor devices. This class of problems was studied extensively by Sethian and Adalsteinsson [15]. The location of the zero level set, before etching starts, is shown in Figure 12. The dimensions of this initial profile are also shown, along with those of the computational domain used in the simulations described below.

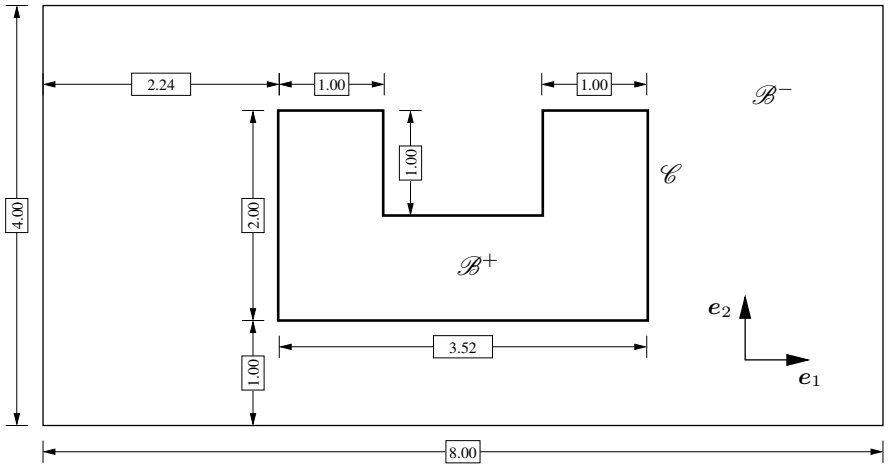


Figure 12. The etching problem: Dimensions of the initial profile and the computational domain.

Chemical etching is the simplest problem in this class, since it corresponds to $\mathrm{v} = 1$. This leads to a constant extensional velocity, $v_\mathrm{e} = 1$, and (6) is satisfied trivially. For this case, we report results obtained with a uniform mesh of bilinear quadrilateral elements with characteristic mesh size $h = 0.02$. The results are shown in Figure 13. Clearly, inward corners remain sharp whereas smooth rarefaction fans develop at outward corners. This indicates convergence to the correct entropy solution, i.e. the solution that satisfies the entropy condition first proposed by Sethian [16] for propagating interfaces and which is similar to that for scalar hyperbolic conservation laws (see also [17]).

The ion milling (sputter etching) process is a more challenging problem to simulate since the rate at which material is etched away from the surface is a function of the angle between the incident ion beam and the normal to the surface. The form of such a *yield function* is often obtained by fitting experimental data [15]. Confining attention to the case where the beam impinges on the surface vertically from above, the angle $\theta(\boldsymbol{x}, t)$ between the beam and the local normal, $\boldsymbol{n}$, to a given level set can be defined as

$$\theta = \cos^{-1}\left(\boldsymbol{n} \cdot (-\boldsymbol{e}_2)\right), \tag{35}$$

where $\boldsymbol{e}_2$ is the unit vector in the vertical direction shown in Figure 12. Following Sethian and co-workers [1, 15], we consider the following yield functions:

$$\hat{\mathrm{v}}_\mathrm{c}(\theta) = \cos(\theta), \qquad \text{(convex)}, \tag{36}$$

$$\hat{\mathrm{v}}_\mathrm{nc}(\theta) = 5\cos(\theta) - 4\cos^3(\theta), \qquad \text{(non-convex)}. \tag{37}$$

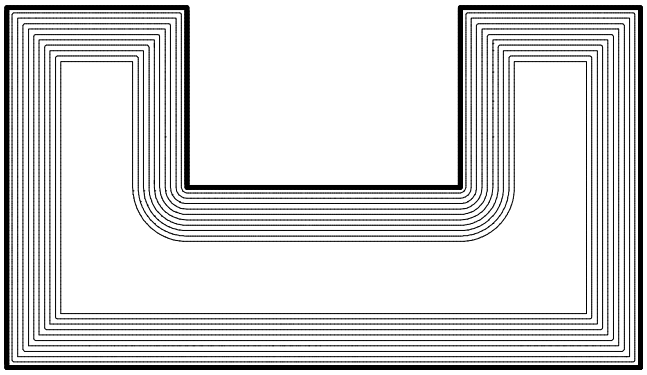


Figure 13. The chemical (isotropic) etching problem.

Note that it is possible to show that $\hat{\mathrm{v}}_{\mathrm{c}}$ and $\hat{\mathrm{v}}_{\mathrm{nc}}$ are both extensional. This can also be inferred from the observation that setting $v_{\mathrm{e}} = \hat{\mathrm{v}}_{\mathrm{c}}(\theta)$ or $v_{\mathrm{e}} = \hat{\mathrm{v}}_{\mathrm{nc}}(\theta)$ everywhere in the domain leads to the propagation of parallel level sets at the same speed. We also note that neither speed function is defined where θ is discontinuous. In the problem under consideration, a discontinuity in the level-set gradient—already present at $t = 0$ in the form of a sharp corner in the profile of Figure 12—consists of a jump in θ between 0 and $\pi/2$. Since $\hat{\mathrm{v}}_{\mathrm{c}}$ is convex over the interval $\theta \in [0, \pi/2]$, it is possible to resort to blending between the extrema $\hat{\mathrm{v}}_{\mathrm{c}}(0)$ and $\hat{\mathrm{v}}_{\mathrm{c}}(\pi/2)$, to evaluate $\hat{\mathrm{v}}_{\mathrm{c}}$ at the discontinuity. However, such an approach can encounter difficulties with non-convex speed functions like $\hat{\mathrm{v}}_{\mathrm{nc}}$; consider for example that the maximum value of $\hat{\mathrm{v}}_{\mathrm{nc}}$ is reached at some intermediate value between 0 and $\pi/2$ (see Sethian and Adalsteinsson [15]). For details regarding more sophisticated strategies—e.g. monotone schemes and ENO/WENO versions thereof—which are widely adopted in problems involving non-smooth solutions, see Zhang and Shu [20] and references therein.

In the convex case, $v_{\mathrm{e}} = \hat{\mathrm{v}}_{\mathrm{c}}(\theta)$, the *original* level set equation (4) takes the form

$$\dot{\phi} - \boldsymbol{e}_2 \cdot \boldsymbol{\nabla}\phi = 0. \tag{38}$$

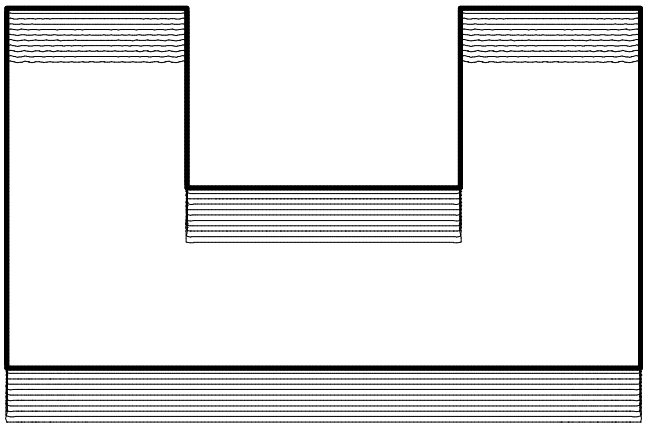
This is a linear pure-advection equation. Given that $-\boldsymbol{e}_2$ is the advective velocity, we expect horizontal portions of the zero level set to migrate downward (visibility effects are ignored allowing the bottom surface to migrate) while vertical ones remain stationary.

It is important to note that the modified equation (10) also yields (38) if the assumption $\|\boldsymbol{\nabla}\phi\| = 1$ is used consistently with (8), such that (35) gives $\theta = \cos^{-1}(-\boldsymbol{\nabla}\phi \cdot \boldsymbol{e}_2)$. Remarkably however, failure to consistently invoke this assumption gives rise to misleading numerical artifacts, as explained below (see Figure 18).

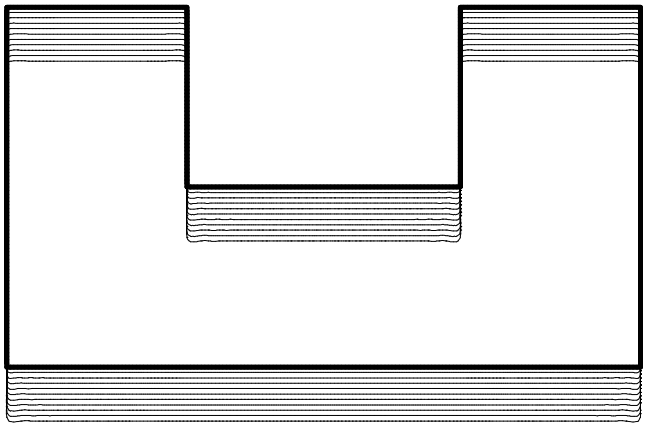
The Bubnov-Galerkin weak form of this problem can be stated as follows: Find $\phi^h_{n+1} \in \mathscr{U}^h$ such that

$$\int_{\mathscr{B}} \phi^h_{n+1} w^h \,\mathrm{dV} = \int_{\mathscr{B}} \phi^h_n w^h \,\mathrm{dV} + \Delta t \int_{\mathscr{B}} \left(\boldsymbol{e}_2 \cdot \boldsymbol{\nabla}\phi^h_n\right) w^h \,\mathrm{dV}, \tag{39}$$

for all $w^h \in \mathscr{U}^h$. To attenuate the spurious spatial oscillations expected to appear in the numerical solution of this advective system, the following stabilized Petrov-Galerkin weak



(a)



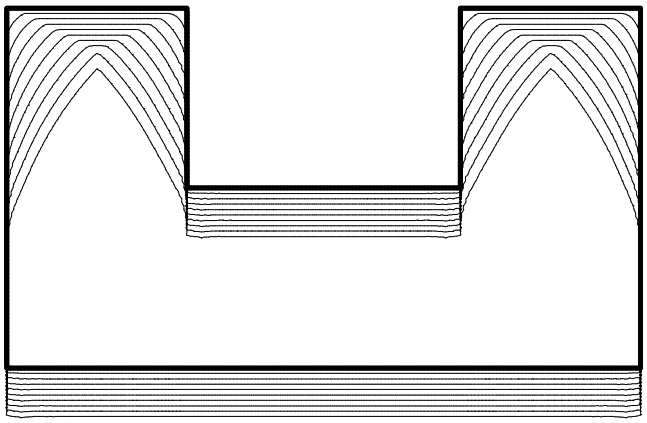
(b)

Figure 14. The ion milling problem with convex speed function, $v_e = \cos(\theta)$. Results shown were obtained with (a) Bubnov-Galerkin formulation (b) Petrov-Galerkin (SUPG) formulation.

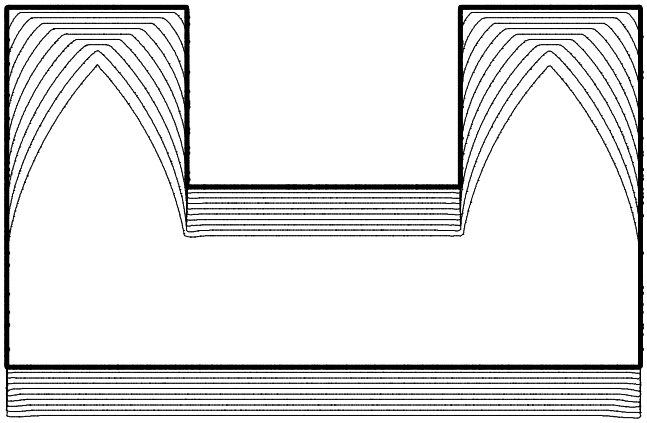
form can be used in place of (39):

$$\sum_{e=1}^{M} \int_{\mathscr{B}^e} \phi_{n+1}^h z^h \, \mathrm{dV} = \sum_{e=1}^{M} \int_{\mathscr{B}^e} \phi_n^h z^h \, \mathrm{dV} + \Delta t \sum_{e=1}^{M} \int_{\mathscr{B}^e} \left(\boldsymbol{e}_2 \cdot \boldsymbol{\nabla} \phi_n^h \right) z^h \, \mathrm{dV}, \tag{40}$$

where $z^h = w^h + \tau^e \left(\boldsymbol{e}_2 \cdot \boldsymbol{\nabla} w^h \right)$ is the perturbed weighting function, $\tau^e = h_2^e / \sqrt{15}$ is the stabilization parameter (see Brooks and Hughes [2]) and h_2^e is the dimension of $\mathscr{B}^e$ in the vertical direction. Forward-Euler time integration is adopted in both formulations, (39) and



(a)



(b)

Figure 15. The ion milling problem with non-convex speed function, $v_e = 5\cos(\theta) - 4\cos^3(\theta)$. Results shown were obtained on (a) Cartesian mesh (b) unstructured locally-refined mesh.

(40), of this problem. A step size $\Delta t = 10^{-3}$ is used with this first-order scheme for added temporal accuracy. Conditional stability is attainable with $\Delta t \leq 0.02$ in this case, since the domain is partitioned with a uniform mesh with characteristic element size $h = 0.02$.

A comparison of results obtained with the *non-stabilized* Bubnov-Galerkin method and the stabilized SUPG algorithm are shown in Figures 14(a) and 14(b), respectively. Despite the presence of discontinuities in $\boldsymbol{\nabla}\phi$ and v_e, the numerical results obtained—even with the *non-stabilized* Bubnov-Galerkin formulation—do not exhibit significant oscillations in the solution.

Minor oscillations in the interface geometry can be observed for both algorithms, though in different regions of the domain.

Finally, we consider the non-convex case, $v_e = \hat{v}_{nc}(\theta)$. Using the assumption $\|\boldsymbol{\nabla}\phi\| = 1$ consistently leads to the governing equation

$$\dot{\phi} - 5(\boldsymbol{e}_2 \cdot \boldsymbol{\nabla}\phi) + 4(\boldsymbol{e}_2 \cdot \boldsymbol{\nabla}\phi)^3 = 0. \tag{41}$$

A Bubnov-Galerkin approach, combined with second-order Runge-Kutta time stepping, leads to a two-stage update procedure analogous to (19)–(20). Figure 15(a) shows results obtained with a uniform mesh of bilinear quadrilaterals ($h = 0.02$), while Figure 15(b) shows results obtained on an unstructured triangulated grid, locally refined in the neighborhood of the zero level set. Noting that

$$\max_{0 \le \theta \le \frac{\pi}{2}} \hat{v}_{nc}(\theta) \approx \hat{v}_{nc}(0.277\,\pi) \approx 2.152,$$

the condition for stability is $\Delta t \lesssim 9.3 \times 10^{-3}$ on the Cartesian mesh and $\Delta t \lesssim 6.2 \times 10^{-3}$ on the unstructured one. For improved accuracy, a step size $\Delta t = 10^{-4}$ is used with both grids. The computations performed on both grids appear to converge to the correct entropy solution. Qualitatively, the results in Figure 15 are in excellent agreement with those of Barth and Sethian [1], obtained using their explicit discontinuity-capturing Petrov-Galerkin formulation. Clearly, the faceting implied by the non-convex yield function is captured, sharp corners are well preserved and the solution has no visible spurious oscillations.

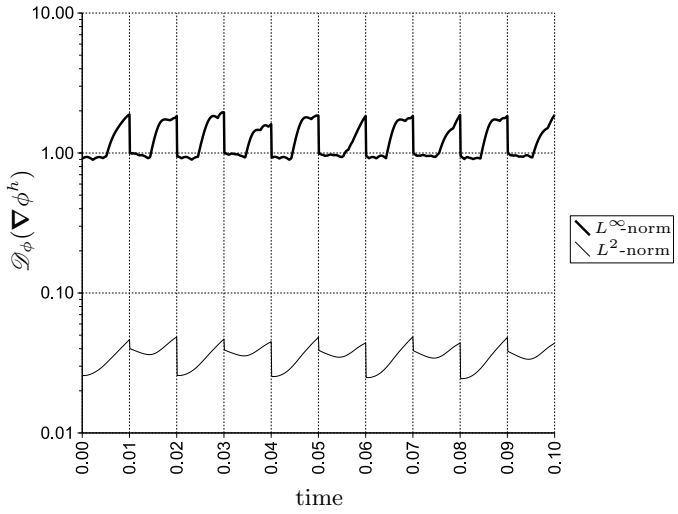


Figure 16. The deviation of $\|\boldsymbol{\nabla}\phi^h\|$ from unity, plotted against simulation time. The effect of reinitialization every 0.01 time units can be seen clearly.

It is important to note that, although reinitialization of the level set field was not needed in any of the previous numerical examples, it was found to be necessary to maintain stability in this case. In all the problems presented before, $\|\boldsymbol{\nabla}\phi^h\| = 1$ is maintained for all $t \in [0, t_f]$. In the present problem however, $\|\boldsymbol{\nabla}\phi^h\|$ deviates from unity as the solution progresses in time, despite the fact that the velocity field is extensional. This fact is illustrated in Figure 16, where the normalized L^2- and L^∞-norms of this deviation, defined as $\mathscr{D}_\phi(\boldsymbol{\nabla}\phi^h) = \|\boldsymbol{\nabla}\phi^h\| - 1$,

are plotted against time. It is clear from the graph that both norms increase with time—though not necessarily in a monotonic fashion—and experience a sudden drop each time the reinitialization algorithm is invoked (every 0.01 units of time in this case). Our numerical experiments suggest that the L^∞-norm of $\mathscr{D}_\phi$ must not be allowed to exceed 2.0 if stability is to be maintained. This issue is clearly one that merits further investigation and will be the subject of future work. At present, we suggest that the L^∞-norm of $\mathscr{D}_\phi$ could be used as a heuristic indicator of when to invoke reinitialization. It is noted that the idea of allowing $\|\boldsymbol{\nabla}\phi^h\|$ to deviate slightly from unity before invoking reinitialization has been widely accepted since the notion of reinitialization was first introduced by Chopp [4]. Moreover, the question of when to invoke reinitialization, and the idea of monitoring the norm of $\mathscr{D}_\phi$ and invoking reinitialization when this norm reaches a given value are discussed by Peng *et al.* [14].

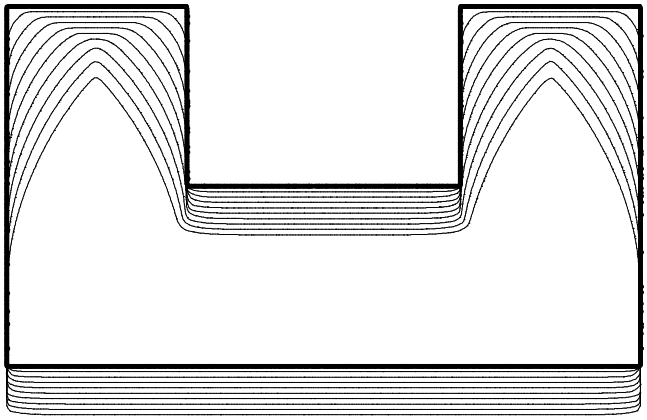


Figure 17. Loss of sharpness at corners due to excessively frequent reinitialization [compare to Figure 15(b)].

In addition to rebuilding a signed-distance function to the interface, the reinitialization algorithm of Section 3.3 introduces some error which has the effect of rounding sharp corners, thereby improving stability. This is the same effect achieved by discontinuity-capturing schemes which introduce artificial diffusion/viscosity terms, e.g. Lax-Friedrichs (see also Hansbo [7]). In the current scheme, the frequency of reinitialization governs the amount of numerical diffusion introduced; i.e. more frequent reinitialization improves numerical stability but introduces more error. Reinitializing too frequently leads to a loss of accuracy at the corners, as can be seen by comparing the results in Figure 17, obtained by reinitializing every single step, to those in Figure 15(b), obtained by reinitializing every 10 time steps. It is also noted that the error introduced by the reinitialization algorithm is due in large part to the treatment of each subsurface $\mathscr{C}^e$, in the current two-dimensional framework, as a rectilinear segment. As explained in Section 3.3, this assumption is better justified, and thus introduces less error, in *linear* triangular elements, than in *bilinear* quadrilateral elements. More frequent reinitialization is therefore needed to maintain stability on triangulated grids; for example, the results shown in Figure 15(b) were obtained on a triangulated grid by reinitializing every 10 time steps, and yet are very similar to those shown in Figure 15(a), obtained on a Cartesian grid by reinitializing every 100 time steps.

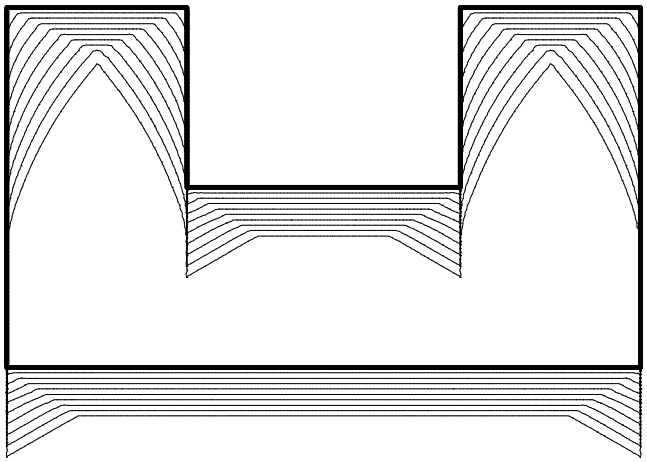


Figure 18. The non-physical faceting artifact [compare to Figure 15(b)] resulting from an inconsistent use of the assumption $\|\boldsymbol{\nabla}\phi\| = 1$.

Furthermore, numerical experiments reveal that the artifact illustrated in Figure 18 is caused by the failure to use the assumption $\|\boldsymbol{\nabla}\phi\| = 1$ consistently. More specifically, the wrong solution is obtained when the aforementioned assumption is used in the original level set equation (4) to obtain the modified equation (10), but is *not* used in (8) to justify writing $\boldsymbol{n} = \boldsymbol{\nabla}\phi$. The resulting (incorrect) governing equation is

$$\dot{\phi} - 5\left(\boldsymbol{e}_2 \cdot \frac{\boldsymbol{\nabla}\phi}{\|\boldsymbol{\nabla}\phi\|}\right) + 4\left(\boldsymbol{e}_2 \cdot \frac{\boldsymbol{\nabla}\phi}{\|\boldsymbol{\nabla}\phi\|}\right)^3 = 0. \tag{42}$$

Expectedly, erroneous results are obtained when $\|\boldsymbol{\nabla}\phi\|$ is evaluated numerically in the neighborhood of discontinuities in $\boldsymbol{\nabla}\phi$. It is perhaps worth noting that Barth and Sethian [1] observed a similar effect—which they described as "unphysical faceting on the lower surface"—in results they obtained using an implicit discontinuity-capturing Petrov-Galerkin formulation based on the original level set equation.

5. SUMMARY AND CONCLUDING REMARKS

In this paper, we presented an approach to simplify the algorithmic treatment of the level set equation for evolving-interface problems. The approach relies on the level set function being sufficiently close to a signed-distance function, and using that assumption consistently. Such a signed-distance function is preserved as the solution progresses in time if the velocity field is extensional. Noting that this can be shown only where the level set field and the extensional velocity field are both sufficiently smooth, we applied our "assumed-gradient" algorithm to various problems in which these smoothness requirements are not met. In these numerical experiments, convergence to the correct entropy solution was observed, even though the proposed scheme did not resort to spatial stabilization and/or discontinuity-capturing terms such as those used by Barth and Sethian [1]. This is particularly remarkable for a class

of problems considered, namely sputter-etching simulations, where the governing equation takes the form of a Hamilton-Jacobi equation with convex or non-convex Hamiltonian. Results obtained in these simulations were comparable, qualitatively, to those obtained using more complex stabilized formulations such as those in [1]. With the exception of the case of a non-convex Hamiltonian, stability was achieved using time step sizes on the order of the standard CFL condition. In the non-convex case, we found that $\|\boldsymbol{\nabla}\phi^h\| = 1$ was quickly violated, but that periodic reinitialization was sufficient to correct this deviation *and* maintain stability.

Importantly, the assumed gradient algorithm described herein has been discretized using structured and unstructured finite-element meshes based on bilinear quadrilateral and linear triangular elements. Particularly for quadrilateral-based meshes, our approach appears to be rather unique. The algorithm is easy to implement and represents a viable alternative to more complex schemes based on Petrov-Galerkin formulations with discontinuity capturing operators, for example. Extensions to higher-order elements and multi-dimensional problems are obviously worth pursuing.

Finally, it should be noted that the algorithms used here to build the extensional velocity field and to reinitialize the level set field are admittedly expensive. As such, when the procedure described herein is applied to structured Cartesian meshes, the overall procedure is not as efficient as more mature schemes employing the latest techniques from the level set community that have been specifically tailored for such cases. However, we contend that other options are available and that the particular techniques used herein are not of central importance to the success of the assumed-gradient formulation. The adaptation of fast marching methods to arbitrary quadrilateral meshes, for example, would allow for the reinitialization and velocity extension algorithms to be greatly improved. The development of such techniques and the extension of this work to higher-order elements are areas of future research.

ACKNOWLEDGEMENTS

The work of H. M. and J. D. at Duke University was supported by Sandia National Laboratories under grant 184592. The work of H. M. and K. G. at the University of Michigan was supported by the National Science Foundation under grant CMS0075989. J. D. would also like to thank Nicolas Moës and N. Sukumar for several helpful comments during the preparation of this manuscript.

REFERENCES

1. Barth TJ, Sethian JA. Numerical schemes for the Hamilton-Jacobi and level set equations on triangulated domains. *Journal of Computational Physics* 1998; **145**(1):1–40.
2. Brooks AN, Hughes TJR. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1982; **32**:199–259.
3. Chessa J, Belytschko T. An enriched finite element method and level sets for axisymmetric two-phase flow with surface tension. *International Journal for Numerical Methods in Engineering* 2003; **58**(13):2041–2064.
4. Chopp, DL. Computing minimal surfaces via level set curvature flow. *Journal of Computational Physics* 1993; **106**(1):77–91.
5. Dolbow JE, Fried E, Ji H. A numerical strategy for investigating the kinetic response of stimulus responsive hydrogels. *Computer Methods in Applied Mechanics and Engineering* 2005; in press.
6. Garikipati K, Rao V. Recent advances in models for thermal oxidation of silicon. *Journal of Computational Physics* 2001; **174**(1):138–170.
7. Hansbo P. Explicit streamline diffusion finite element methods for the compressible Euler equations in conservation variables. *Journal of Computational Physics* 1993; **109**(2):274–288.

8. Ji H, Mourad HM, Fried E, Dolbow JE. Kinetics of thermally-induced swelling of hydrogels. *International Journal of Solids and Structures* 2005; in press.
9. Johnson C. *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Cambridge University Press: Cambridge, 1987.
10. Malladi R, Sethian JA, Vemuri BC. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1995; **17**(2):158–175.
11. Mourad HM, Garikipati K. Advances in the numerical treatment of grain boundary migration: Coupling with mass transport and mechanics. *Journal of Computational Physics*; submitted. Also at http://www.arxiv.org/abs/physics/0411012.
12. Osher S, Sethian JA. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulation. *Journal of Computational Physics* 1988; **79**(12):12–49.
13. Osher SJ, Fedkiw RP. *Level Set Methods and Dynamic Implicit Surfaces.* Springer-Verlag: New York, 2002.
14. Peng D, Merriman B, Osher S, Zhao H, Kang M. A PDE-based fast local level set method. *Journal of Computational Physics* 1999; **155**(2):410–438.
15. Sethian JA, Adalsteinsson D. An overview of level set methods for etching, deposition, and lithography development. *IEEE Transactions on Semiconductor Manufacturing* 1997; **10**(1):167–184.
16. Sethian JA. *An Analysis of Flame Propagation.* Ph.D. Dissertation. University of California, Berkeley, 1982.
17. Sethian JA. *Level Set Methods and Fast Marching Methods.* Cambridge University Press: New York, 1999.
18. Sethian JA, Vladimirsky, A. Ordered upwind methods for static Hamilton-Jacobi equations: Theory and algorithms. *SIAM Journal on Numerical Analysis*, 2003; **41**(1):325–363.
19. Sussman M, Fatemi E. An efficient, interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM Journal of Scientific Computing* 1999; **20**(4):1165–1191.
20. Zhang Y-T, Shu C-W. High-order WENO schemes for Hamilton-Jacobi equations on triangular meshes. *SIAM Journal of Scientific Computing* 2003; **24**(3):1005–1030.
21. Zhao H-K, Chan T, Merriman B, Osher S. A variational level set approach to multiphase motion. *Journal of Computational Physics* 1996; **127**(1):179–195.